Struct
(Kiểu Cấu Trúc)

Trịnh Tấn Đạt
Khoa CNTT - Đại Học Sài Gòn
Email: trinhtandat@sgu.edu.vn
Website: https://sites.google.com/site/ttdat88/

Nội dung

- Đặt vấn đề
- Định nghĩa kiểu cấu trúc (Struct)
- Khai báo biến cấu trúc
- Truy xuất các thành phần trong cấu trúc
- Mảng cấu trúc
- Các ví dụ minh họa
- Bài Tập

Đặt vấn đề

❖ Thông tin 1 SV

- MSSV : kiểu chuỗi
- Tên SV : kiểu chuỗi
- Ngày sinh: kiểu chuỗi
- Giới tính: ký tự (F/M)
- Điểm Toán, Lý, Hóa : số thực

❖ Yêu cầu

- Lưu thông tin của n SV?
- Truyền thông tin của n SV vào hàm?

Đặt vấn đề

❖ Khai báo các biến để lưu trữ 1 SV

- char mssv[7]; // “0123456”
- char hoten[30]; // “Nguyen Van A”
- char ntns[8]; // “01/01/1980”
- char phai; // ‘m’ ⇔ Nam, ‘f’ ⇔ Nữ
- float toan, ly, hoa; // 8.5 9.0 10.0

❖ Truyền thông tin 1 SV cho hàm

- void xuat(char mssv[], char hoten[], char ntns[], char phai, float toan, float ly, float hoa);

Đặt vấn đề

❖ Nhận xét

- Đặt tên biến khó khăn và khó quản lý
- Truyền tham số cho hàm quá nhiều
- Tìm kiếm, sắp xếp, sao chép,… khó khăn
- Tốn nhiều bộ nhớ
- …

❖ Ý tưởng

- Gom những thông tin của cùng 1 SV thành một kiểu dữ liệu mới => Kiểu struct

Định nghĩa kiểu cấu trúc

- Struct (Cấu trúc) : là kiểu dữ liệu gồm một nhóm các thành phần có kiểu không giống nhau, mỗi thành phần được xác định bằng một tên riêng biệt.

- Kiểu dữ liệu của mỗi thành phần trong cấu trúc là một kiểu đã được định nghĩa trước , kể cả mảng và các cấu trúc khác.

Khai báo kiểu cấu trúc

- Một kiểu cấu trúc được định nghĩa với từ khóa **struct**.

```
typedef   struct  <TênKiểu>
{
    <Kiểu dữ liệu thành phần>      <Tên thành phần>;
    <Kiểu dữ liệu thành phần>      <Tên thành phần>;
    <Kiểu dữ liệu thành phần>      <Tên thành phần>;
    <Kiểu dữ liệu thành phần>      <Tên thành phần>;
    ....
};
```

Ví dụ: Khai Báo kiểu cấu trúc

```
typedef  struct  TDate
{
      int  day;
      int  month;
     int  year;
};
```

```
typedef  struct  TBook
{
      char     title[100];
      char     author[50];
      float     price;
};
```

```
typedef  struct  Point3D
{
      float x;
      float y;
      float z;
};
```

```
typedef  struct  TStudent
{
      char     ID[10];
      char     firstname[10];
      char     lastname[30];
      TDate   dob;
      float     mark[10];
};
```

```
typedef  struct  Point2D
{
      float x;
      float y;
};
```

```
typedef  struct  PhanSo
{
      int  tuso;
      int  mauso;
};
```

Khai báo biến kiểu cấu trúc

- Cú pháp: **struct <tên kiểu cấu trúc> <tên biến cấu trúc>;**

(trong **C++ có thể bỏ struct**)

- Nếu dùng typedef: **<tên kiểu cấu trúc> <tên biến cấu trúc>;**
- **Ví dụ:** Khai báo biến đơn kiểu cấu trúc

```
typedef struct Point2D
{
    float x;
    float y;
};
// khai báo biến P1,P2
Point2D P1,P2;
```

```
typedef struct PhanSo
{
    int tuso;
    int mauso;
};

PhanSo Ps1,Ps2;
```

Khai báo biến kiểu cấu trúc

- Khai báo mảng cấu trúc: cũng tương tự khai báo biến mảng trên các kiểu dữ liệu khác .
- Cú pháp: **<tên kiểu cấu trúc> <tên biến mảng cấu trúc>[HẰNG_SỐ];**

```
typedef struct Point2D
{
    float x;
    float y;
};
// khai báo biến P1,P2
Point2D P1,P2;
// mảng cấu trúc
Point2D ArrayP[100];
```

```
typedef struct PhanSo
{
    int tuso;
    int mauso;
};

PhanSo Ps1,Ps2;
//mảng phân số
PhanSo Ps[50];
```

Khởi tạo giá trị cho biến cấu trúc

- Cú pháp:

  <tênkiểucấutrúc> <tênbiến> = {<giá trị 1>,…,<giá trị n>};

- **Ví dụ:**

```
Point2D P1 = {1.5,0.5},P2 = {2,4};
PhanSo  Ps1 = {1,2};
PhanSo  Ps2 = {3,4};
```

Truy xuất các thành phần trong struct

❖ Đặc điểm

- Không thể truy xuất trực tiếp
- Truy xuất thông qua toán tử thành phần cấu trúc . hay còn gọi là toán tử chấm (dot operation) và tên thành phần
- Cú pháp: **<tên biến cấu trúc>.<tên thành phần>**

Ví dụ:

```
typedef struct Point2D
{
    float x;
    float y;
};
Point2D  P1={1.5,0.5},P2={2,4};
cout << P1.x << " "<< P1.y << endl;
cout << P2.x << " "<< P2.y << endl;
```

Truy xuất các thành phần trong struct

Ví dụ

```
#include <iostream>
using namespace std;
typedef struct Point2D
{
	float x;
	float y;
};
int main()
{
	Point2D P1 = {2,4}, P2;
	cout<<"diem P1 la:(" << P1.x <<","<< P1.y <<")"<<endl;
	cout <<"nhap vao diem P2:";
	cin>> P2.x >> P2.y;
	cout<<"diem P2 la:(" << P2.x <<","<< P2.y <<")"<<endl;
	return 0;
}
```

Truy xuất các thành phần trong struct

- Mảng cấu trúc: `<tên biến mảng cấu trúc>[chỉ số].<tên thành phần>`

Ví dụ:

```
typedef struct Point2D
{
    float x;
    float y;
};
Point2D  P[30];
P[0].x = 10;
P[0].y = 4;
P[1].x = 1.5;
P[1].y = 6.5;
```

Truy xuất các thành phần trong struct

Ví dụ

```
#include <iostream>
using namespace std;
typedef struct Point2D
{
	float x;
	float y;
};
int main()
{
	Point2D P[30];
	cout<<"nhap diem thu nhat:";
	cin>>P[0].x>>P[0].y;
	cout<<"diem P0 la:(" << P[0].x <<","<< P[0].y <<")"<<endl;
	cout<<"nhap diem thu hai:";
	cin>>P[1].x>>P[1].y;
	cout<<"diem P1 la:(" << P[1].x <<","<< P[1].y <<")"<<endl;
	return 0;
}
```

Ví dụ

```
#include <iostream>
using namespace std;
typedef struct Point2D
{
	float x;
	float y;
};
int main()
{
	int n;
	Point2D Q[30];
	// nhap vao n diem trong mang Q;
	cout<<"nhap vao so luong diem;";
	cin>>n;
	for(int i=0;i<n;i++)
	{
		cout << "nhap vao hoanh do:"; cin>>Q[i].x;
		cout << "nhap vao tung do:"; cin>>Q[i].y;
	}
	// in ra n diem
	for(int i=0;i<n;i++)
	{
		cout<<"diem Q[" << i << "] la:(" << Q[i].x <<","<< Q[i].y <<")"<<endl;
	}
	return 0;
}
```

Ví dụ: Nhập xuất danh sách nhân viên

```
/* Danh sach nhan vien */
#include <iostream>
#include <stdlib.h>
using namespace std;
typedef struct TDate
{
    int ngay;
    int thang;
    int nam;
};
typedef struct nhanvien
{
    int manv;
    char hoten[30];
    TDate dob;
};
```

Lưu ý:

Thành phần của cấu trúc là cấu trúc khác
Thành phần cấu trúc là biến mảng

```
int main()
{       nhanvien snv[50];
        int i, n;
        cout<<"Nhap vao luong so nhan vien:";
        cin>>n;
        //Nhap danh sach nhan vien
        for(i = 0; i <n; i++)
        {
                cout<<"Nhap vao ma nhan vien thu "<< i + 1 << ":";
                cin>>snv[i].manv;
                cout<<"nhap vao ngay sinh:";
                cin>>snv[i].dob.ngay>>snv[i].dob.thang>>snv[i].dob.nam;
                cout<<"Nhap vao ho ten: ";
                fflush(stdin);  // ko co dong nay ????
                gets(snv[i].hoten); // gets dugn nhap chuoi ky tu
        }
        //in danh sach nhan vien ...
```

Ví dụ: Nhập xuất danh sách nhân viên

```
for(i = 0; i<n; i++)
        {
                cout<<snv[i].manv <<endl;
                cout<<snv[i].dob.ngay <<"/" << snv[i].dob.thang <<"/"<<snv[i].dob.nam  <<endl;
                cout<<snv[i].hoten <<endl << endl;
        }
         return 0;
}
```

Gán dữ liệu cho biến cấu trúc

- Có 2 cách

```
<biến cấu trúc đích> = <biến cấu trúc nguồn>;

<biến cấu trúc đích>.<tên thành phần> = <giá trị>;
```

```
typedef struct Point2D
{
    float x;
    float y;
};
Point2D  P1={1.5,0.5},P2, P3;
P2 = P1;
P3.x = P1.x *2;
P3.y = P1.y *4;
```

Kích thước kiểu cấu trúc

- Toán tử sizeof()

```
struct A
{
    int a;
    double b;
};
sizeof(A) = ???
```

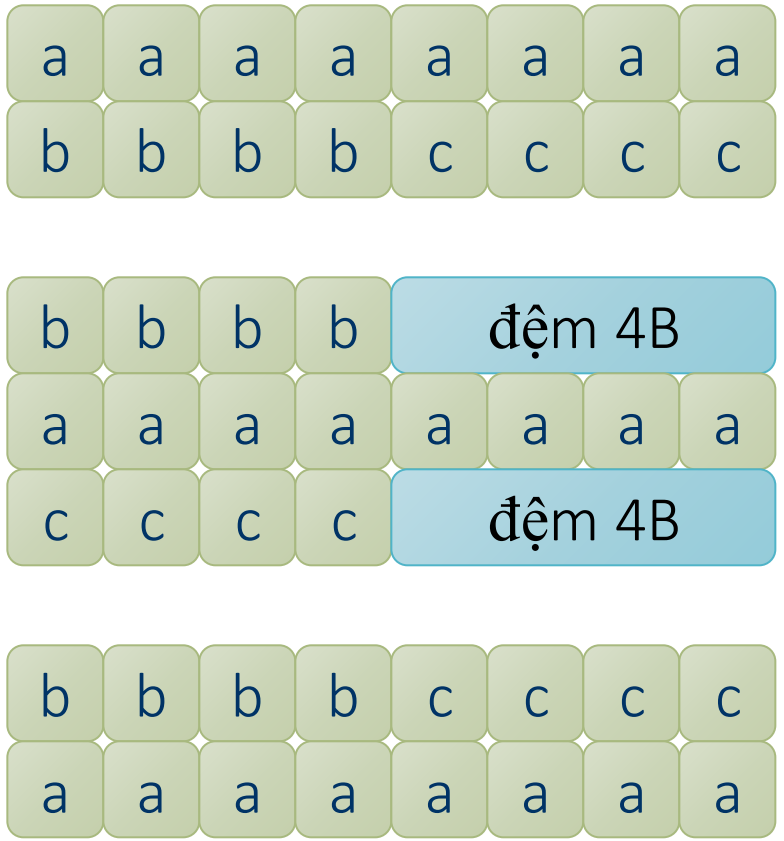
```
struct B1
{
    int a;
    int b;
    double c;
};
sizeof(B1) = ???
```

```
struct B2
{
    int a;
    double c;
    int b;
};
sizeof(B2) = ???
```

Kích thước kiểu cấu trúc

- Ví dụ

```
struct A {
    double a;
    int b;
    int c;
};
struct B {
    int b;
    double a;
    int c;
};
struct C {
    int b;
    int c;
    double a;
};
```



Các lưu ý về cấu trúc

- Kiểu cấu trúc được định nghĩa để làm khuôn dạng còn biến cấu trúc được khai báo để sử dụng khuôn dạng đã định nghĩa.
- Trong C++, có thể bỏ từ khóa struct khi khai báo biến (hoặc sử dụng typedef)
- Khi nhập các biến kiểu số thực trong cấu trúc phải nhập thông qua một biến trung gian.

```
typedef struct DIEM
{
    float x, y;
};
DIEM d1;
float temp;
cin>>temp;
d1.x = temp;
```

Các lưu ý về cấu trúc

❖ Truyền cấu trúc cho hàm

- Giống như truyền kiểu dữ liệu cơ sở
  - Tham trị (không thay đổi sau khi kết thúc hàm)
  - Tham chiếu (thay đổi sau khi kết thúc hàm)
- Ví dụ

```
typedef struct DIEM
{
    int x, y;
};
void xuat1(int x, int y) { … };
void xuat2(DIEM diem) { … };
void xuat3(DIEM &diem) { … };
```

Tự tìm hiểu thêm (không bắt buộc)

- Kiểu union
- Kiểu enum

Ví dụ minh họa

- Viết chương trình cộng hai phân số trong đó các tử số và mẫu số là các số nguyên. Yêu cầu kết quả phân số ở dạng tối giản.

```
#include <iostream>
using namespace std;
typedef struct PHANSO
{
	int tuso;
	int mauso;
};
// prototype
void nhapps(PHANSO &ps);
void inps (PHANSO ps);
int USCLN (int a, int b);
void toigianps(PHANSO &ps);
PHANSO congps (PHANSO ps1,PHANSO ps2);
```

Ví dụ minh họa

```
int main()
{
	PHANSO ps1, ps2, tongps;
	nhapps(ps1);
	toigianps(ps1);
	inps(ps1);
	nhapps(ps2);
	toigianps(ps2);
	inps(ps2);
	tongps = congps(ps1,ps2);
	toigianps(tongps);
	cout<<"tong 2 phan so la:";
	inps(tongps);
	return 0;
}
```

Ví dụ minh họa

```
void nhapps(PHANSO &ps)
{
	cout<<"nhap tu so:";
	cin>>ps.tuso;
	cout<<"nhap mau so:";
	cin>>ps.mauso;
}
```

```
void inps(PHANSO ps)
{
	cout<<"Phan so la:" <<ps.tuso <<"/"<<ps.mauso<<endl;
}
```

Ví dụ minh họa

```
int USCLN(int a, int b)
{
	int r = a%b;
	while(r!=0)
	{
		a = b;
		b = r;
		r = a%b;
	}
	return b;
}
```

```
void toigianps(PHANSO &ps)
{
	int uocchung = USCLN(ps.tuso,ps.mauso);
	ps.tuso = ps.tuso/uocchung;
	ps.mauso = ps.mauso/uocchung;
}
```

```
PHANSO congps(PHANSO ps1, PHANSO ps2)
{
	PHANSO kq;
	kq.tuso = ps1.tuso*ps2.mauso+ps2.tuso*ps1.mauso;
	kq.mauso = ps1.mauso*ps2.mauso;
	return kq;
}
```

Bài Tập

1) Viết chương trình nhập vào thông tin của một sinh viên, xuất thông tin của sinh viên vừa nhập ra màn hình.

   Cho biết thông tin một sinh viên bao gồm: mã sinh viên (số nguyên), họ và tên SV (chuỗi ký tự), giới tính (chuỗi ký tự), địa chỉ (chuỗi ký tự), điểm toán, lý hóa (số thực)

- Lưu ý: dùng hàm gets() để nhập chuỗi

  trước khi nhập chuỗi nhớ xóa vùng điệm dùng fflush(stdin);

Ví dụ: char hoten[30];

```
fflush(stdin);
gets(hoten);
```

Bài Tập

2) Phân số

- Khai báo kiểu dữ liệu phân số (PHANSO)
- Nhập/Xuất phân số
- Rút gọn phân số
- Tính tổng, hiệu, tích, thương hai phân số
- Kiểm tra phân số tối giản
- Quy đồng hai phân số
- Kiểm tra phân số âm hay dương
- So sánh hai phân số

Bài Tập

3) Điểm trong mặt phẳng Oxy

- Khai báo kiểu dữ liệu điểm (DIEM)
- Nhập/Xuất tọa độ điểm
- Tính khoảng cách giữa hai điểm
- Tìm điểm đối xứng qua gốc toạ độ/trục Ox/Oy
- Kiểm tra điểm thuộc phần tư nào?

Bài Tập

4) Tam giác

- Khai báo kiểu dữ liệu tam giác (TAMGIAC)
- Nhập/Xuất tam giác
- Tính chu vi, diện tích tam giác

Bài Tập: Mảng cấu trúc (điểm)

1) Trong mặt phẳng tọa độ *OXY* cho *n* điểm có tọa độ x_i, y_i (i=1..*n*).

   a. Đếm số lượng điểm nằm ở phần tư thứ I (x>0 và y>0)

   b. Tìm cặp điểm gần nhau nhất

Bài Tập: Mảng cấu trúc (phân số)

2)

Cho mảng một chiều các phân số (tử số và mẫu số là số nguyên), hãy thực hiện các công việc sau đây:

a. In các phân số của mảng ở dạng tối giản.

b. Đếm xem trong mảng có bao nhiêu phân số có giá trị lớn hơn 0 và nhỏ hơn 1.

c. Đếm xem trong mảng có bao nhiêu phân số (ở dạng tối giản) mà tử số và mẫu số của phân số đó đều là các số nguyên tố.

d. Viết hàm sắp xếp các phân số có giá trị tăng dần.

e. Tính tổng giá trị các phân số.

Bài Tập: Mảng cấu trúc (Sinh Viên)

3) Trong kỳ thi tuyển sinh cao học có *n* thí sinh tham gia; mỗi thí sinh cần quản lý các thông tin sau:

- Số báo danh (kiểu chuỗi, 8 ký tự)
- Họ và tên (kiểu chuỗi, tối đa 36 ký tự)
- Điểm *môn ngoại ngữ* (số nguyên)
- Điểm *môn cơ sở* (số nguyên)
- Điểm *môn chuyên ngành* (số nguyên)

a. Nhập thông tin *n* thí sinh theo mô tả trên.

b. Tìm những thí sinh có điểm *môn chuyên ngành* là cao nhất.

c. Tìm những thí sinh có điểm thi *môn ngoại ngữ* ≥ 5 và có tổng điểm *môn cơ sở* và *môn chuyên ngành* ≥ 12.